04

Especificación de los servlets

Transcripción

[00:00] Hola a todos y todas, bienvenidas y bienvenidos una vez más a este nuevo video sobre servlet. Vean que nosotros conseguimos deployar nuestra aplicación en un servidor diferente de Tomcat. Hicimos todo dentro de eclipse para ser deployado después en Tomcat y vean que ahora fue bastante fácil la transición de Tomcat para Jetty. Lo que estuvo difícil fue la configuración de Jetty en sí mismo.

[00:27] Pero el deploy de nuestra aplicación en Jetty fue muy fácil. ¿Y por qué pudimos hacer eso? Gracias a que nosotros nunca codamos, nunca estuvimos desarrollando específicamente para un servlet container en particular, sino que nosotros siempre estuvimos dependiendo de interfaces, entonces vean que acá tenemos las interfaces que nosotros hemos estado usando.

[00:56] Dependemos de nuestro filter, que es una interfaz, nuestro Servlet nosotros en realidad implementamos de httpServlet, pero es una clase genérica. Entonces también pudimos abstraernos del tipo de servlet container que deployamos, ServletResponse, ServletRequest, y HttpServletRequest y response.

[01:21] Todas fueron interfaces. Entonces, nosotros solamente dependíamos de contratos o sea, qué cosa deberíamos implementar, pero no dependíamos de un código en concreto. Las interfaces solamente tienen contratos, no una implementación de un código. Entonces esas interfaces como no ejecutan código, nosotros pudimos hacer esa dependencia.

[01:52] Y ahora, si esas interfaces no tienen la implementación del código, ¿quién tiene esa implementación? Quien tiene la implementación son nuestros servlets containers. El servlet container de Tomcat tendrá una especificación para, o sea, no una especificación, sino que también obedece ese contrato de la interfaz y lo implementa de una forma que tal vez el servlet containers de Jetty lo hace de otra forma.

[02:18] Sin embargo, los dos están cumpliendo ese contrato de tener los métodos adecuados. Si ahora nos vamos, por ejemplo a Tomcat vean que nosotros, dentro de nuestro Apache Tomcat, dentro de la carpeta lib tenemos un montón de .jars, que ahí tiene un montón de implementaciones, pero nuestra aplicación solamente depende de ese servlet-api.jar.

[02:44] Solamente ese es suficiente para que nosotros podamos deployar una aplicación web, o por lo menos nuestra aplicación web. Lo mismo, por ejemplo, con Jetty. Jetty-home si vamos a lib, Jetty tiene acá un jetty-jakarta-servlet-api. Los otros .jar no sé lo que hacen, pero sé que ese en particula, me sirve a mí para poder crear esos servlets.

[03:17] Entonces, mientras nosotros coloquemos un nuestra aplicación en un lugar donde exista ese .jar, nosotros vamos a poder ejecutar ese Servlet. Y todo eso es gracias a una especificación de jakarta EE. Jakarta EE es una especificación que nos permite hacer todos esos servlets y cualquier servlet container que tenga la especificación Jakarta EE va a soportar nuestra aplicación. Entonces conseguimos enviarla de un servlet container a otro.

[03:49] Y otras especificaciones, por ejemplo, son JPA, JDBC, que es para la parte de bases de datos, entonces todas ellas tienen interfaces de las cuales nosotros podemos hacer una implement, implementar esa interface y con eso, bueno, desarrollar código utilizando esa especificación. Y voy a probarles por ejemplo, que nosotros solamente dependemos de ese servlet-api.

[04:19] Entonces si nosotros vamos acá a nuestro código, bueno tenemos ese httpServlet que viene de ese servlet-api que les había mostrado y yo, por ejemplo si voy a gerenciador, clic derecho propiedades, me voy a Java build path, dentro de ese classPath van a ver que tienen este Server runtime de Apache Tomcat.

[04:45] Acá están todos esos .jar que trae consigo nuestro Tomcat. Entonces voy a eliminar esos .jar, apply and close y vean lo siguiente, ahora nos está dando error HttpServlet no está siendo encontrado, porque nosotros eliminamos todas esas librerías que tenía nuestro Apache Tomcat.

[05:11] Pero por ejemplo, si yo me voy ahora de nuevo a properties, propiedades de nuestro gerenciador Java Build Path, clico en classPath y agregó una external jar, podemos ir a nuestro Apache Tomcat, versión lib y acá si yo agrego solamente a servlet-api, voy a colocar un apply and close, vean que los errores se fueron. ¿Por qué?

[05:38] Porque esos httpServlet están en ese .jar en específico, que les había mostrado acá, en ese servlet-api. Y entonces vean que esto nos permite eso que les había dicho que en cualquier lugar, cualquier servlet container que tenga ese .jar vamos a poder ejecutar nuestra aplicación web.

[06:03] Entonces con esto nosotros ya tenemos una aplicación bien armada, hemos conseguido deployarlo en servlet containers diferentes y con esto los veo en el próximo video. Nos vemos allá.